Advanced Regression: Introduction to non-linear regression

Garyfallos Konstantinoudis

Feb 28, 2025

Non-linear regression

So far we have studied the linear regression models, which are very flexible tool for estimating relationships in a set of data. In particular, we have seen:

- Model set up, fitting, and residual diagnostics.
- Model building and model comparison.
- ANOVA/ likelihood ratio test.
- Random intercepts and random slopes.

> Note
>
> One of the most crucial assumptions of the previous lecture was that the associations between the outcome and the covariates are linear. What if the association was not linear?

A clarification

In the name normal (or Poisson, binomial, etc.) linear model, the word 'linear' refers to the response being modelled as a linear combination of covariates, i.e.

$$Y_i \sim N(\beta_1 + \beta_2 X_{i2} + \cdots + \beta_p X_{ip}, \sigma^2)$$

It does not refer to each covariate-response relationship being linear. Therefore the following is also within the class of normal linear models

$$Y_i \sim N(\beta_1 + \beta_2 X_i + \beta_3 X_i^2, \sigma^2)$$

Here, the relationship between Y and x is quadratic, but it is still a linear model.

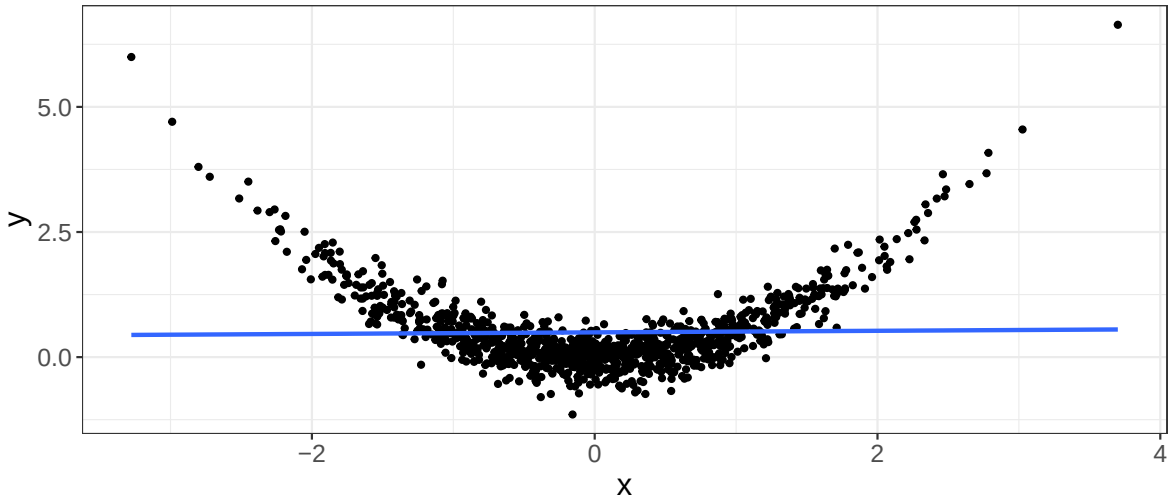
So how do you tell if the relationship is linear?

- Plot each covariate against the response and see what shape the relationship is.
- Make sure you plot it with the transformation used in the model.
- If linear, fit a linear model, if not start thinking of the shape of the relationship.
- Fit as simple a model as possible, do not overcomplicate it unnecessarily. (Occam's razor)

Example

Associations are not solely linear

```
library(dplyr)
library(ggplot2)

set.seed(11)
x <- rnorm(n = 1000)
y <- 0.5 * x * x + rnorm(1000, sd = 0.3)

ggplot(data = data.frame(x = x, y = y), aes(x = x, y = y)) +
  geom_point(cex = 1.2) +
  theme_bw() +
  geom_smooth(method = "lm", formula = y ~ x, se = FALSE, linewidth = 1) +
  theme(text = element_text(size = 15))
```



Basis Expansions

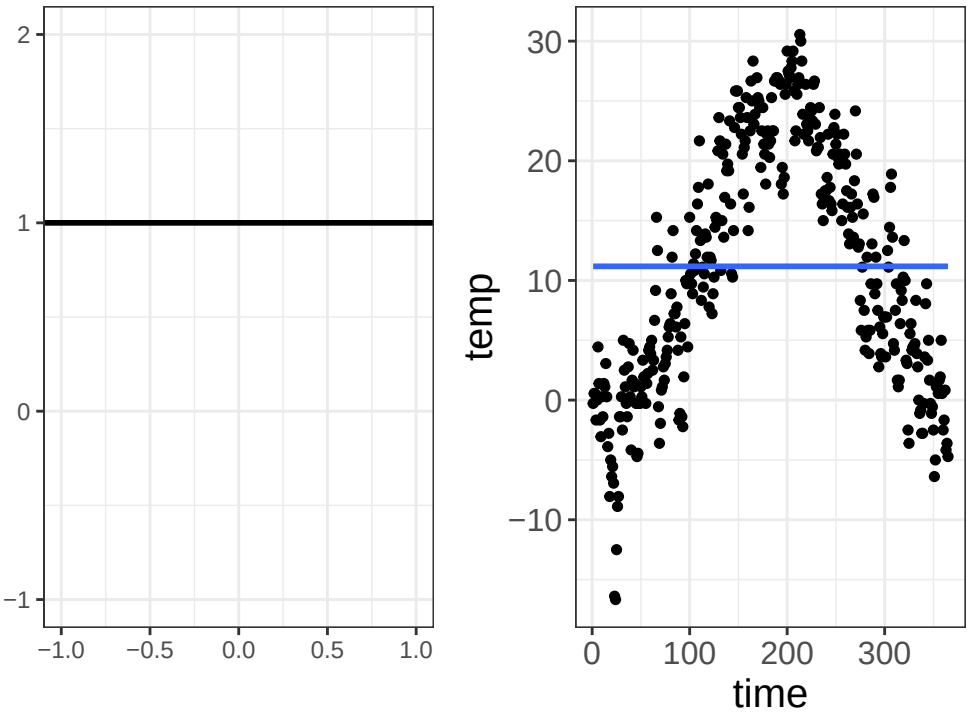
We need to define a set of flexible functions that could capture relationships that are not linear. In general, we can write:

$$Y_i = \sum_{j}^{K} \beta_j \phi_j(X_j) + \epsilon_i$$

which we can write: $f(X) = \beta^T \Phi(X)$ and we say that $\Phi(X)$ is a basis system for f.

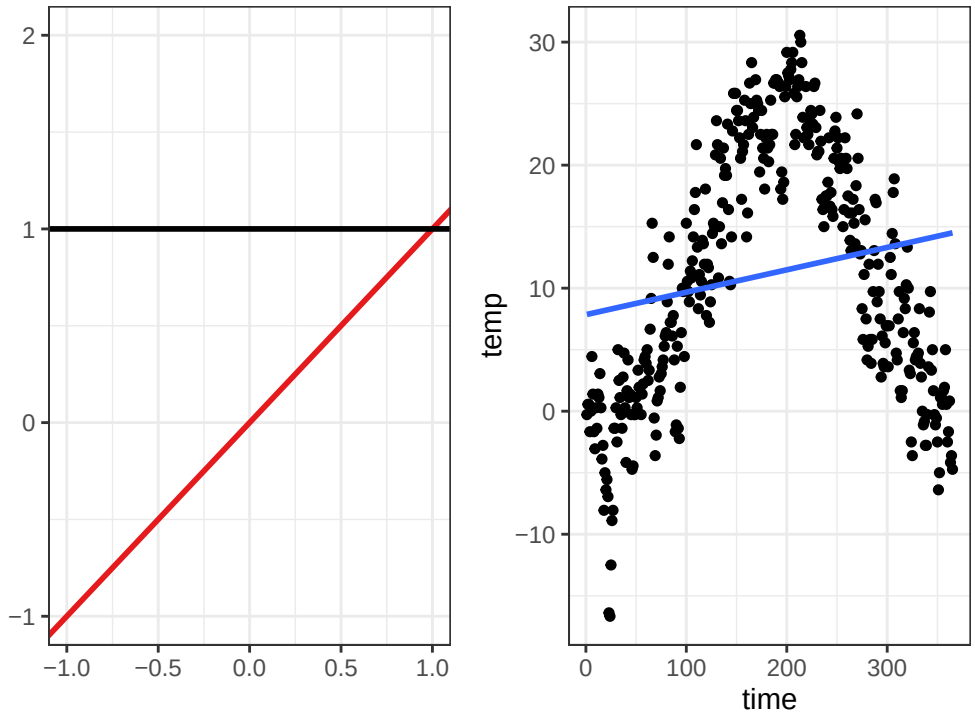
The polynomial basis function

$$\Phi(X) = (1) \text{ ,thus } Y_i = \beta_0 + \epsilon_i$$



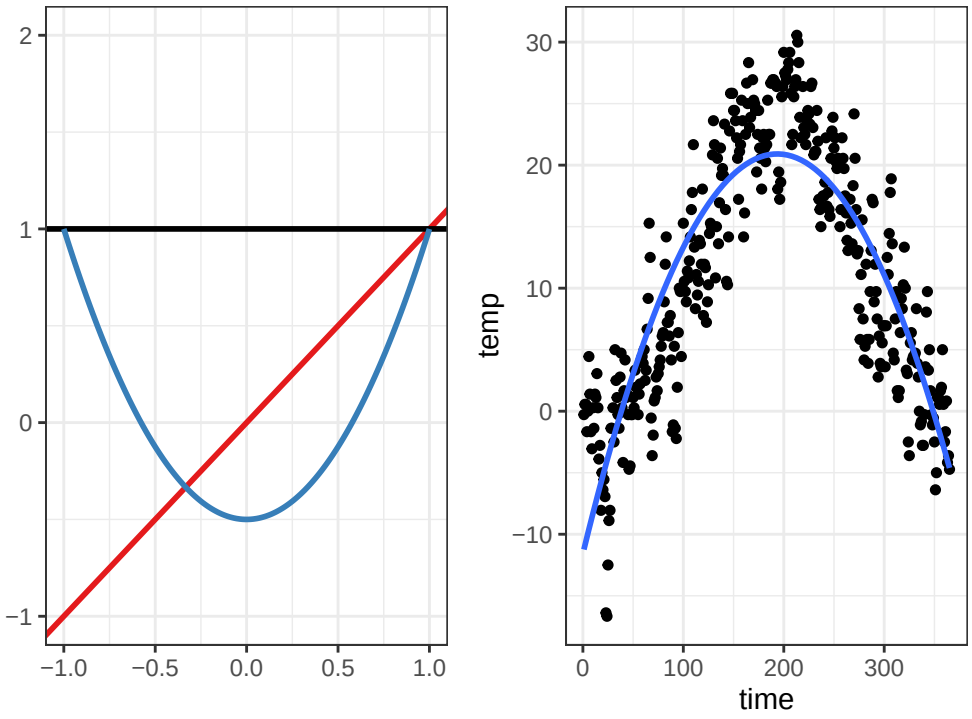
The polynomial basis function

$$\Phi(X) = (1X) \text{ ,thus } Y_i = \beta_0 + \beta_1 X_1 + \epsilon_i$$



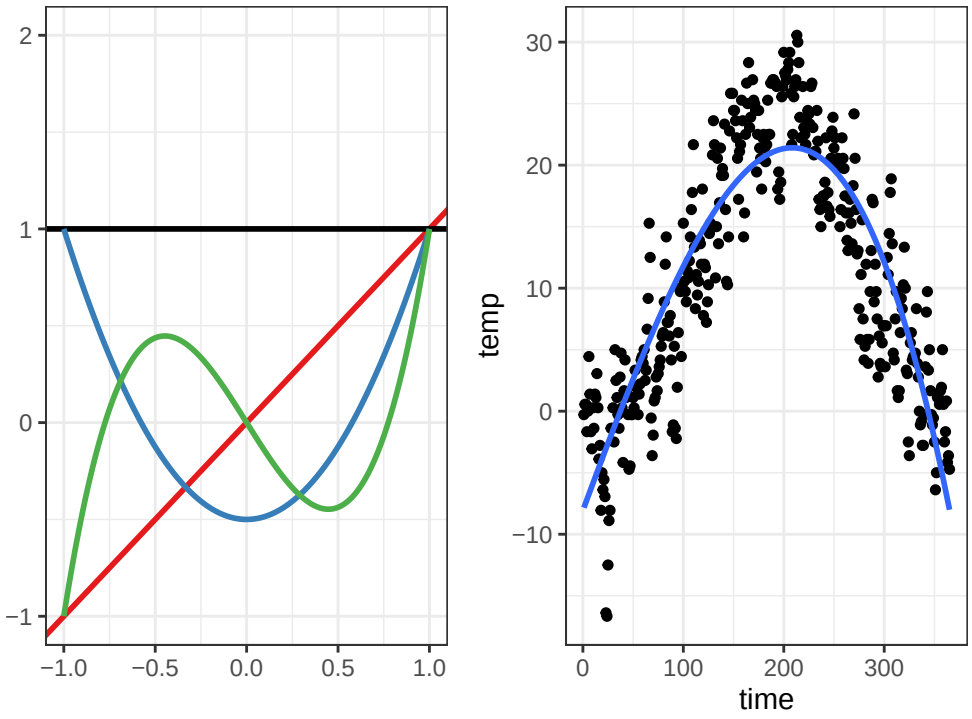
The polynomial basis function

$$\Phi(X) = (1 \; X \; X^2) \text{ ,thus } Y_i = \beta_0 + \beta_1 X + \beta_2 X^2 + \epsilon_i$$



The polynomial basis function

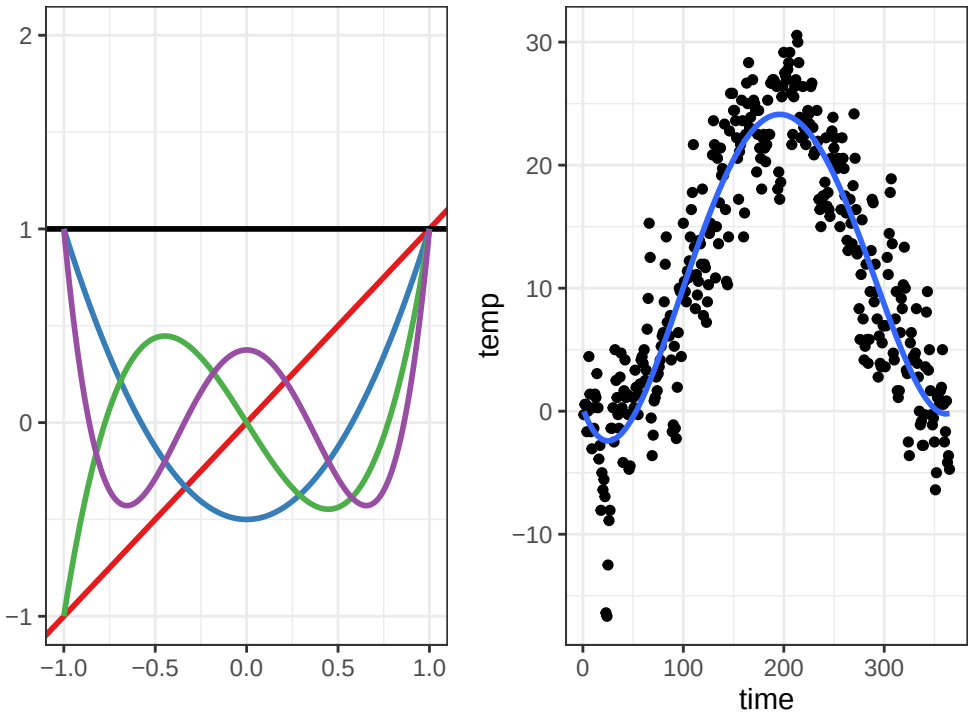
$$\Phi(X) = (1 \; X \; X^2 \; X^3) \text{ ,thus } Y_i = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \epsilon_i$$



The polynomial basis function

$$\Phi(X) = (1 \; X \; X^2 \; X^3 \; X^4) \text{ ,thus}$$

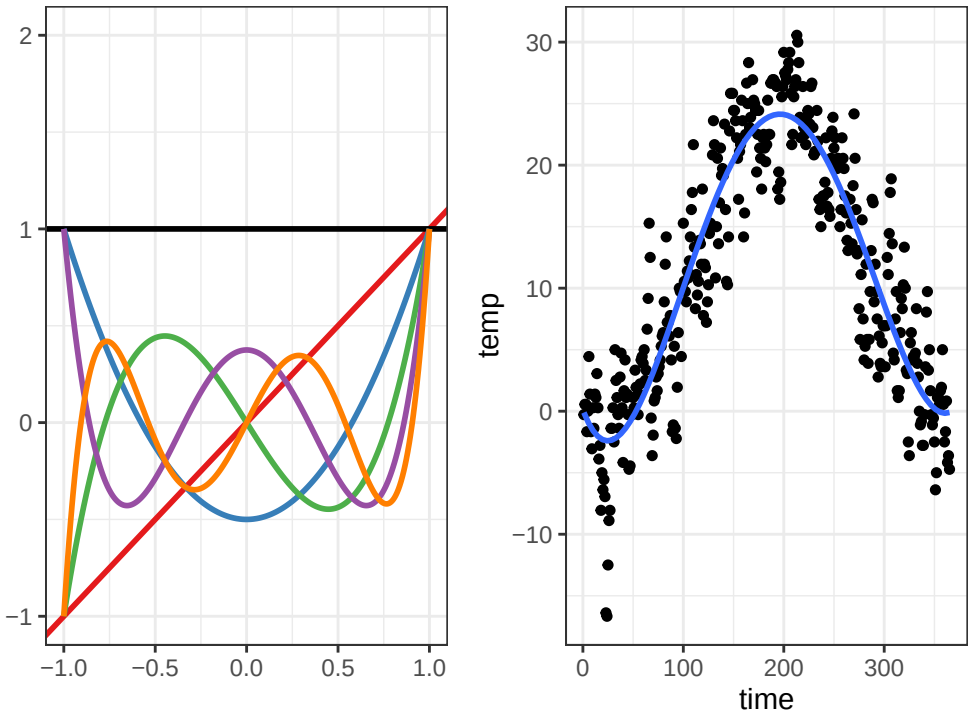
$$Y_i = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \beta_4 X^4 + \epsilon_i$$



The polynomial basis function

$$\Phi(X) = (1 \; X \; X^2 \; X^3 \; X^4 \; X^5) \text{ ,thus}$$

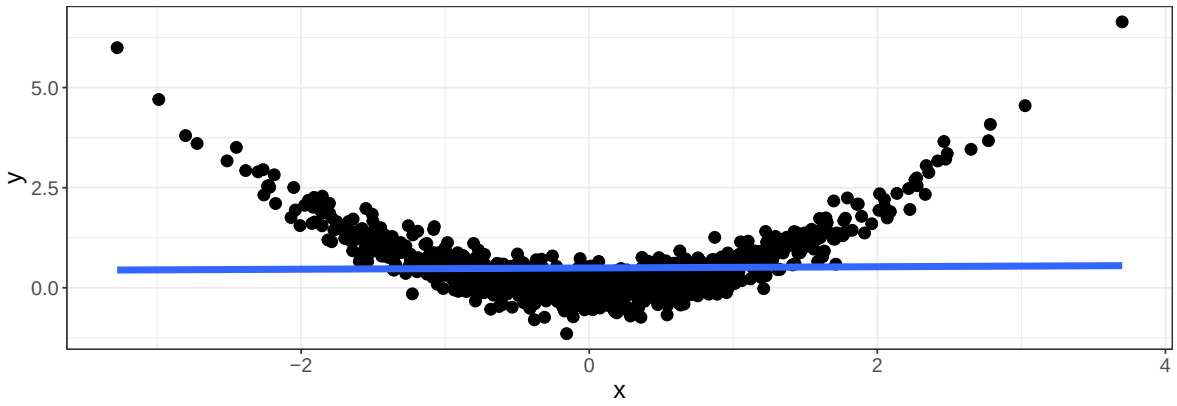
$$Y_i = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \beta_4 X^4 + \beta_5 X^5 + \epsilon_i$$



Example

```
Call:
lm(formula = y ~ x)

Residuals:
    Min      1Q  Median      3Q     Max
-1.6386 -0.4645 -0.1876  0.2583  6.0883

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.49486    0.02471   20.03   <2e-16 ***
x            0.01564    0.02481    0.63    0.529
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7813 on 998 degrees of freedom
Multiple R-squared:  0.000398,  Adjusted R-squared:  -0.0006036
```

```
F-statistic: 0.3973 on 1 and 998 DF,  p-value: 0.5286
```



Fit a polynomial in R

```
lm(y ~ x + I(x^2) + I(x^3)) %>% summary()
```

```
Call:
lm(formula = y ~ x + I(x^2) + I(x^3))

Residuals:
     Min       1Q   Median       3Q      Max
-1.14539 -0.20239  0.01008  0.20206  0.95031

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.012156   0.011541  -1.053    0.292
x            0.007367   0.014781   0.498    0.618
I(x^2)       0.511645   0.006724  76.096   <2e-16 ***
I(x^3)      -0.004603   0.003786  -1.216    0.224
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2982 on 996 degrees of freedom
Multiple R-squared:  0.8547,    Adjusted R-squared:  0.8542
F-statistic:  1952 on 3 and 996 DF,  p-value: < 2.2e-16
```

How can we test the linearity assumption

- Notice that the linear model is nested in the quadratic form
- The same holds for cubic vs quadratic vs linear

```
mod1 <- lm(y ~ x + I(x^2))
mod2 <- lm(y ~ x + I(x^2) + I(x^3))
anova(mod1, mod2)
```

```
Analysis of Variance Table

Model 1: y ~ x + I(x^2)
Model 2: y ~ x + I(x^2) + I(x^3)
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1    997 88.702
2    996 88.571  1   0.13147 1.4784 0.2243
```

Pros and cons

Pros

- These curves are quite flexible—a quadratic can fit most biologically plausible curves
- The curves only use 1(quadratic) or 2(cubic) degrees of freedom more than linear, unlike dummy variable models
- The results are not sensitive to choice of boundaries (there aren't any)
- Outliers mostly influence the extremes of the curve, not the center part

Cons

- They use more degrees of freedom than linear, and therefore have less power
- There is still some sensitivity to influential observations

Other types of basis function

- **Harmonics** - The Fourier basis function

$$1, sin(\omega X), cos(\omega X), sin(2\omega X), cos(2\omega X), \dots,$$

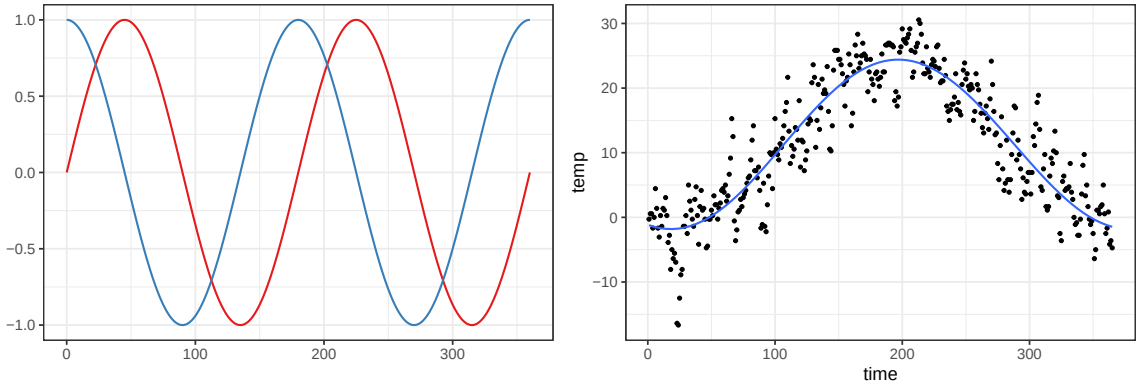
$$sin(m\omega X), cos(m\omega X)$$

- constant ω defines the period of oscillation of the first sine/cosine pair. This is $\omega = 2\pi/P$ where P is the period.

Example: Fourier basis function

$$\Phi(X) = (1\ sin(\omega X)\ cos(\omega X)\ sin(2\omega X)\ cos(2\omega X))\ \text{,thus}$$

$$Y_i = \beta_0 + \beta_1 sin(\omega X) + \beta_2 cos(\omega X) + \beta_3 sin(2\omega X) + \beta_4 cos(2\omega X) + \epsilon_i$$



Pros and cons

Pros

- Excellent computational properties, especially if the observations are equally spaced.
- Natural for describing periodic data, such as the annual weather cycle
- The results are not sensitive to choice of boundaries (there aren't any)

Cons

- functions are periodic; this can be a problem if the data are, for example, growth curves.

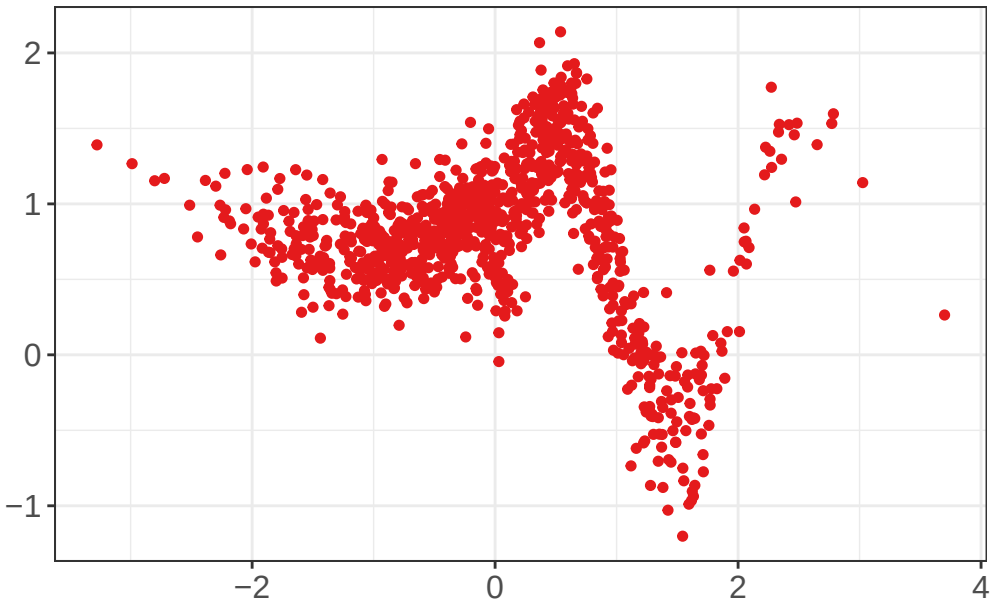
Parametric non-linear effects

- The following models captured the non-linear relationships using simple non-linear functions.
- These types of models are to be preferred if possible, because again they are simpler to make inference from, e.g. the relationship is quadratic.
- However, there may be times when the relationship being modelled does not look like a parametric function. Then what should you do?

Parametric non-linear effects

Consider the following form

```
set.seed(11)
x <- rnorm(n = 1000)
y <- numeric(1000)
y[x < 0] <- 1 + 0.5 * x[x < 0] + 0.2 * x[x < 0]^2 + rnorm(n = length(x[x < 0]), sd = 0.2)
y[x >= 0] <- 0.5 + sin(2 * pi / 2 * x[x >= 0]) + rnorm(n = length(x[x > 0]), sd = 0.3)

ggplot() +
  geom_point(aes(x = x, y = y), cex = 1.3, col = cols[1]) +
  theme_bw() +
  xlab("") +
  ylab("") +
  theme(text = element_text(size = 15))
```



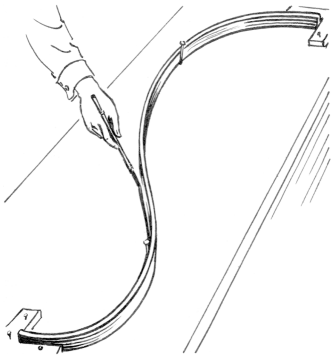
It doesn't look like any simple parametric form (e.g. polynomial, sinusoidal, etc), so what do you do?

Smooth functions

- There are many methods to estimate non-linear relationships such as that on the previous slide.
- They are generically called smooth functions, and include splines (lots of different types), kernel smoothers and local linear smoothers.
- We will focus on splines in this lab, because they are simple to understand graphically and are easy to fit to the data.

Splines

- Splines were originally thin splints of wood used to trace complex, smooth curves in engineering and architectural.
- Splines were pinned to the drawing at points where the spline changed its curvature.



Piecewise Linear Splines

- We begin fitting splines by dividing the range of exposure into pieces.
- Instead of fitting a constant in each piece, we fit a separate linear term in each piece.
- This has more power since it allows variation within categories to predict variation in outcome.
- We can use fewer categories to capture the deviation from a strictly linear curve because we have slopes within category.

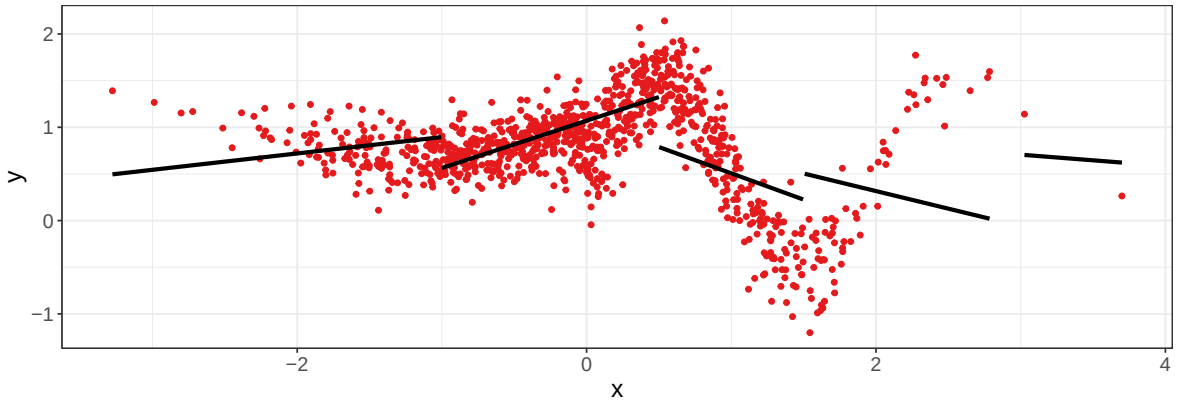
Linear threshold in R

```
dat <- data.frame(x = x, y = y)

lm(y ~ x:(x <= -1) + x:(x <= 0.5) + x:(x <= 1.5) + x:(x <= 3), data = dat) -> mod1
mod1 %>% summary()
```

```
Call:
lm(formula = y ~ x:(x <= -1) + x:(x <= 0.5) + x:(x <= 1.5) +
    x:(x <= 3), data = dat)

Residuals:
     Min       1Q   Median       3Q      Max
-1.68966 -0.23877 -0.01464  0.22753  1.57659

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)     1.07015    0.02093  51.138  < 2e-16 ***
x:x <= -1FALSE -0.12120    0.09429  -1.285  0.19895
```

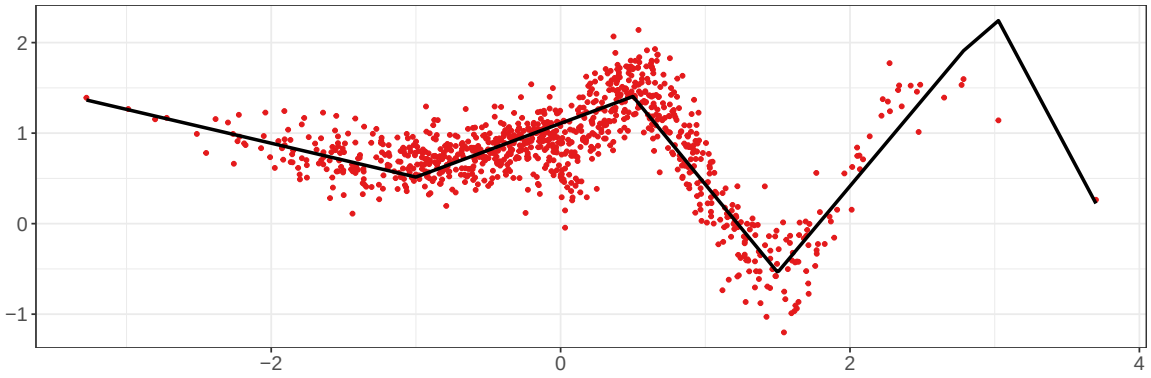
```
x:x <= -1TRUE  -0.45275    0.10641  -4.255 2.29e-05 ***
x:x <= 0.5TRUE  1.07042    0.06596  16.228  < 2e-16 ***
x:x <= 1.5TRUE -0.18660    0.04292  -4.348 1.52e-05 ***
x:x <= 3TRUE   -0.25576    0.09853  -2.596  0.00957 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4497 on 994 degrees of freedom
Multiple R-squared:  0.269, Adjusted R-squared:  0.2653
F-statistic: 73.14 on 5 and 994 DF,  p-value: < 2.2e-16
```

Linear threshold in R



Linear splines in R

```r
library(splines)
fit <- lm(y ~ bs(x, degree = 1, knots = c(-1, 0.5, 1.5, 3)), data = dat)

ggplot() +
  geom_point(aes(x = x, y = y), col = cols[1], cex = 1) +
  theme_bw() +
  xlab("") +
  ylab("") +
  geom_line(aes(x = dat$x, y = fit$fitted.values), lwd = 1) +
  theme(text = element_text(size = 15))
```



Linear splines

Linear threshold model

$$Y_i = \beta_0^{(1)} + \beta_1^{(1)} x + \epsilon_i \qquad x \leq -1 \tag{1}$$
$$Y_i = \beta_0^{(2)} + \beta_1^{(2)} x + \epsilon_i \qquad -1 < x \leq 0.5 \tag{2}$$
$$Y_i = \beta_0^{(3)} + \beta_1^{(3)} x + \epsilon_i \qquad 0.5 < x \leq 1.5 \tag{3}$$
$$Y_i = \beta_0^{(4)} + \beta_1^{(4)} x + \epsilon_i \qquad 1.5 < x \leq 3 \tag{4}$$
$$Y_i = \beta_0^{(5)} + \beta_1^{(5)} x + \epsilon_i \qquad x \geq 3 \tag{5}$$
$$\tag{6}$$

Linear spline model

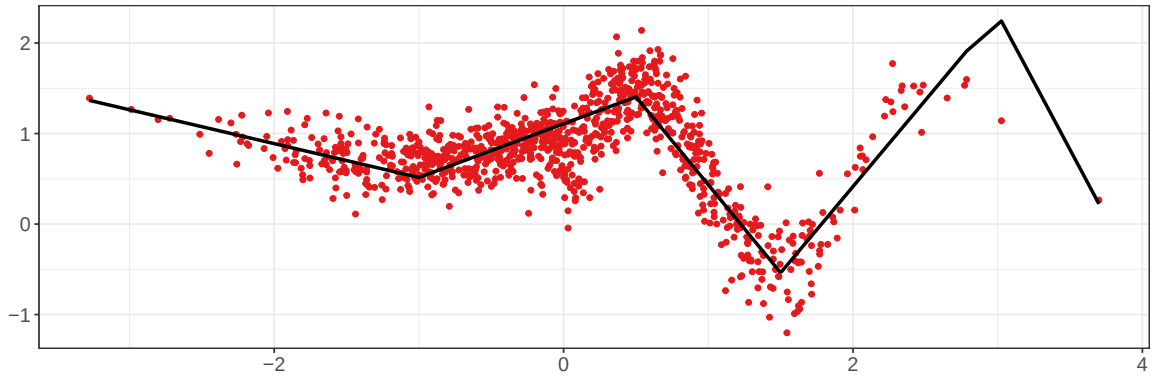
$$Y_i = \beta_0 + \beta_1 x + \beta_2 (x+1)_+ + \beta_3 (x-0.5)_+ + \beta_4 (x-1.5)_+ + \beta_5 (x-3)_+ + \epsilon_i$$

$$(x-k)_+ = \begin{cases} 0, & x < k \\ x-k, & x \geq k \end{cases}$$

Linear splines in R without a package

```
lm(y ~ x + I((x + 1) * (x >= -1)) + I((x - 0.5) * (x >= 0.5)) + I((x - 1.5) * (x >= 1.5)) + I

ggplot() +
  geom_point(aes(x = x, y = y), cex = 1.3, col = cols[1]) +
```

```
  theme_bw() +
  xlab("") +
  ylab("") +
  geom_line(aes(x = dat$x, y = mod2$fitted.values), lwd = 1) +
  theme(text = element_text(size = 15))
```



Cubic splines

Similarly we can define higher order polynomial splines, for instance:

$$Y_i = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \tag{7}$$
$$\beta_4 (X+1)_+ + \beta_5 (X+1)_+^2 + \beta_6 (X+1)_+^3 + \tag{8}$$
$$\beta_7 (x-0.5)_+ + \beta_8 (x-0.5)_+^2 + \beta_9 (x-0.5)_+^3 + \tag{9}$$
$$\cdots + \epsilon_i \tag{10}$$
$$(x-k)_+ = \begin{cases} 0, & x < k \\ x-k, & x \geq k \end{cases} \tag{11}$$

which reduces to the following to ensure smooth curvature on the knots (it can be seen after deriving the first and second derivative):

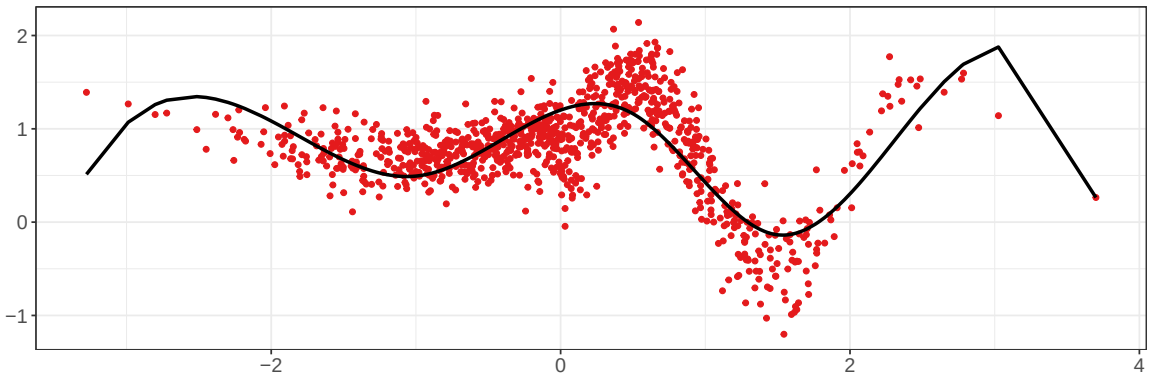
$$Y_i = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x_3 + \tag{12}$$
$$\beta_6 (x+1)_+^3 + \beta_9 (x-0.5)_+^3 + \cdots + \epsilon_i \tag{13}$$

Cubic splines in R

```
fit_c <- lm(y ~ bs(x, degree = 3, knots = c(-1, 0.5, 1.5, 3)), data = dat)
```

without the splines package:

```
fit_nopack <- lm(
  y ~ x + I(x^2) + I(x^3) + I((x + 1)^3 * (x >= -1)) +
    I((x - 0.5)^3 * (x >= 0.5)) + I((x - 1.5)^3 * (x >= 1.5)) + I((x - 3)^3 * (x >= 3)),
  data = dat
)
```



Splines

A spline of order n is a piecewise polynomial function of degree $n-1$ in a variable x.

(Basis) Splines can be:

- Piecewise constant.
- Linear.
- Quadratic.
- Cubic.
- higher order polynomials.
- etc.

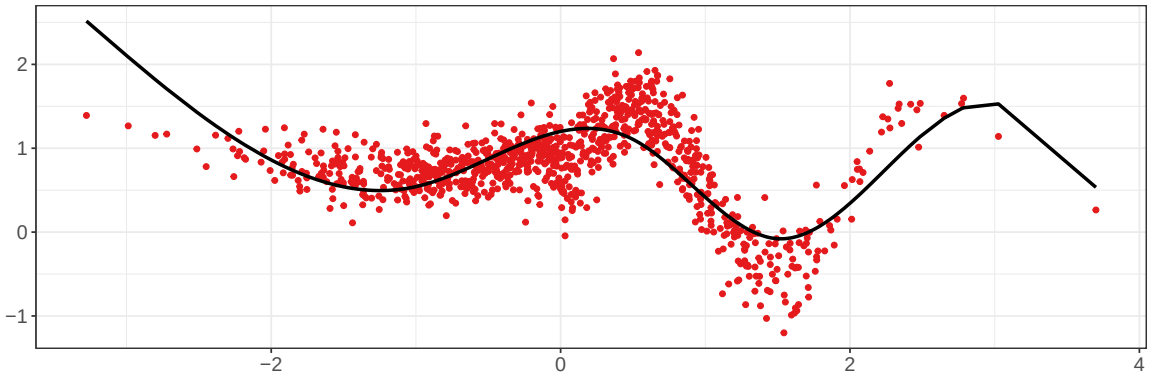
Why are the borders so wiggly?

Natural cubic splines

- Splines can have high variance in the boundaries.
- This problem aggravates if knots too few.
- The natural spline constraints the fit to be linear before and after the first and last knots.
- This stabilises the fitting and makes a more reasonable assumption.

Natural cubic splines in R

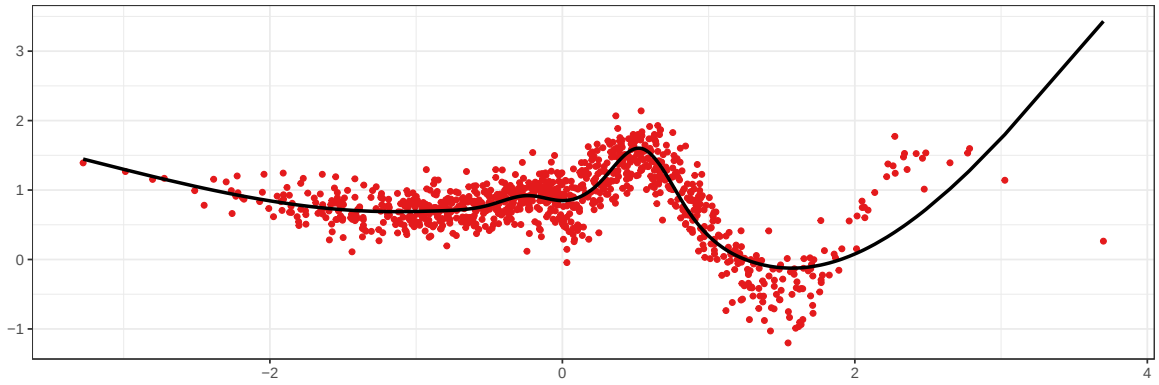
```
fit_ns <- lm(y ~ ns(x, knots = c(-1, 0.5, 1.5, 3)), data = dat)

ggplot() +
  geom_point(aes(x = x, y = y), cex = 1.3, col = cols[1]) +
  theme_bw() +
  xlab("") +
  ylab("") +
  geom_line(aes(x = dat$x, y = fit_ns$fitted.values), lwd = 1) +
  theme(text = element_text(size = 15))
```



Well, better fit be achieved with more knots

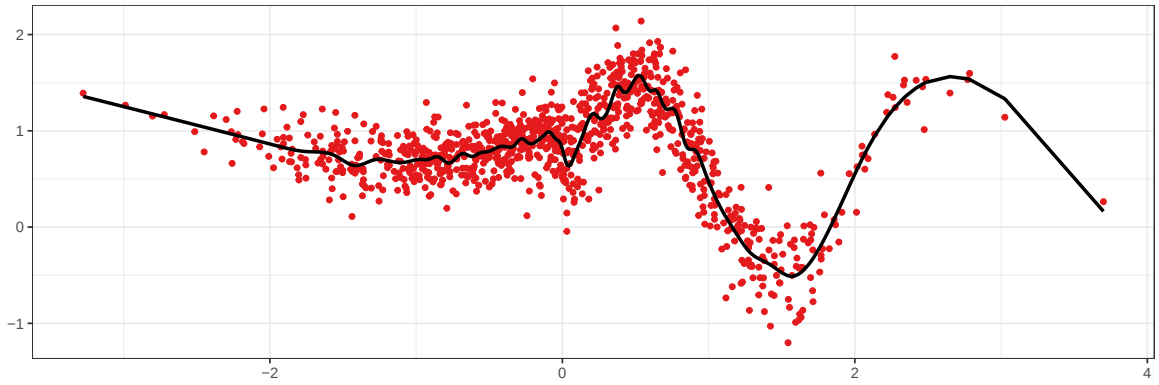
```
fit_ns2 <- lm(y ~ ns(x, df = 10), data = dat)

ggplot() +
```

```
  geom_point(aes(x = x, y = y), cex = 1.3, col = cols[1]) +
  theme_bw() +
  xlab("") +
  ylab("") +
  geom_line(aes(x = dat$x, y = fit_ns2$fitted.values), lwd = 1)
```



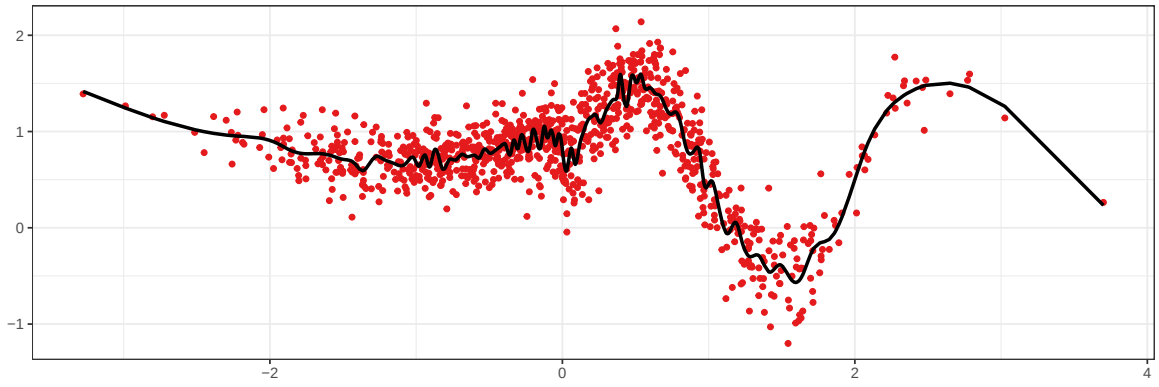
... and more

```
fit_ns2 <- lm(y ~ ns(x, df = 50), data = dat)

ggplot() +
  geom_point(aes(x = x, y = y), cex = 1.3, col = cols[1]) +
  theme_bw() +
  xlab("") +
  ylab("") +
  geom_line(aes(x = dat$x, y = fit_ns2$fitted.values), lwd = 1)
```



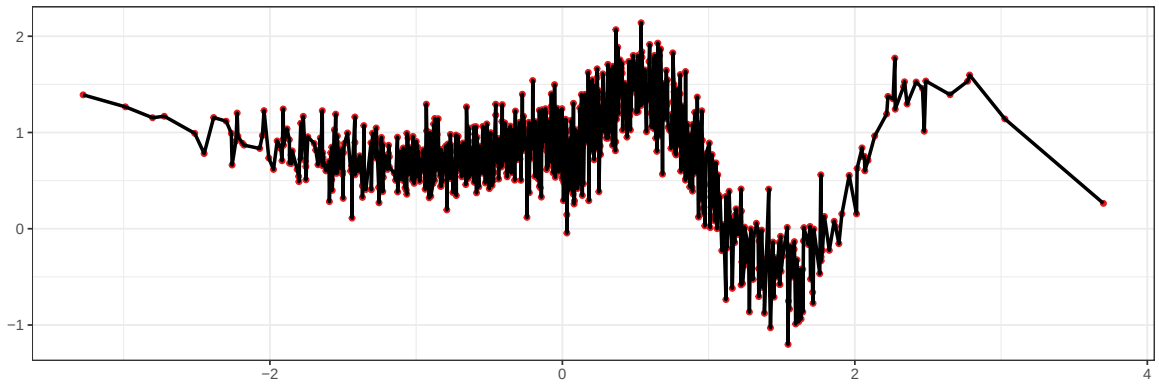
... and more

```
fit_ns3 <- lm(y ~ ns(x, df = 100), data = dat)

ggplot() +
  geom_point(aes(x = x, y = y), cex = 1.3, col = cols[1]) +
  theme_bw() +
  xlab("") +
  ylab("") +
  geom_line(aes(x = dat$x, y = fit_ns3$fitted.values), lwd = 1)
```



... and more

```r
fit_ns4 <- lm(y ~ ns(x, df = 1000), data = dat)

ggplot() +
  geom_point(aes(x = x, y = y), cex = 1.3, col = cols[1]) +
  theme_bw() +
  xlab("") +
  ylab("") +
  geom_line(aes(x = dat$x, y = fit_ns4$fitted.values), lwd = 1)
```



is this a useful model?

Summary

- Introduction to different basis function and how to fit them in R
- Theory and application of linear splines
- Understand more flexible non-parametric functions (focus on splines)

Questions?